

# QSM368ZP-WF&SG368Z 系列

## Linux&Ubuntu 摄像头驱动开发指导

智能模块系列

版本: 1.0

日期: 2024-07-10

状态: 受控文件



上海移远通信技术股份有限公司（以下简称“移远通信”）始终以为客户提供最及时、最全面的服务为宗旨。如需任何帮助，请随时联系我司上海总部，联系方式如下：

上海移远通信技术股份有限公司  
上海市闵行区田林路 1016 号科技绿洲 3 期（B 区）5 号楼 邮编：200233  
电话：+86 21 5108 6236 邮箱：[info@quectel.com](mailto:info@quectel.com)

或联系我司当地办事处，详情请登录：<http://www.quectel.com/cn/support/sales.htm>。

如需技术支持或反馈我司技术文档中的问题，请随时登录网址：  
<http://www.quectel.com/cn/support/technical.htm> 或发送邮件至：[support@quectel.com](mailto:support@quectel.com)。

## 前言

移远通信提供该文档内容以支持客户的产品设计。客户须按照文档中提供的规范、参数来设计产品。同时，您理解并同意，移远通信提供的参考设计仅作为示例。您同意在设计您目标产品时使用您独立的分析、评估和判断。在使用本文档所指导的任何硬软件或服务之前，请仔细阅读本声明。您在此承认并同意，尽管移远通信采取了商业范围内的合理努力来提供尽可能好的体验，但本文档和其所涉及服务是在“可用”基础上提供给您的。移远通信可在未事先通知的情况下，自行决定随时增加、修改或重述本文档。

## 使用和披露限制

### 许可协议

除非移远通信特别授权，否则我司所提供硬软件、材料和文档的接收方须对接收的内容保密，不得将其用于除本项目的实施与开展以外的任何其他目的。

### 版权声明

移远通信产品和本协议项下的第三方产品可能包含受移远通信或第三方材料、硬软件和文档版权保护的相关资料。除非事先得到书面同意，否则您不得获取、使用、向第三方披露我司所提供的文档和信息，或对此类受版权保护的资料进行复制、转载、抄袭、出版、展示、翻译、分发、合并、修改，或创造其衍生作品。移远通信或第三方对受版权保护的资料拥有专有权，不授予或转让任何专利、版权、商标或服务商标权的许可。为避免歧义，除了正常的非独家、免版税的产品使用许可，任何形式的购买都不可被视为授予许可。对于任何违反保密义务、未经授权使用或以其他非法形式恶意使用所述文档和信息的违法侵权行为，移远通信有权追究法律责任。

### 商标

除另行规定，本文档中的任何内容均不授予在广告、宣传或其他方面使用移远通信或第三方的任何商标、商号及名称，或其缩略语，或其仿冒品的权利。

### 第三方权利

您理解本文档可能涉及一个或多个属于第三方的硬软件和文档（“第三方材料”）。您对此类第三方材料的使用应受本文档的所有限制和义务约束。

移远通信针对第三方材料不做任何明示或暗示的保证或陈述，包括但不限于任何暗示或法定的适销性或特定用途的适用性、平静受益权、系统集成、信息准确性以及与许可技术或被许可人使用许可技术相关的不侵犯任何第三方知识产权的保证。本协议中的任何内容都不构成移远通信对任何移远通信产品或任何其他硬件、设备、工具、信息或产品的开发、增强、修改、分销、营销、销售、提供销售或以其他方式维持生产的陈述或保证。此外，移远通信免除因交易过程、使用或贸易而产生的任何和所有保证。

## 隐私声明

为实现移远通信产品功能，特定设备数据将会上传至移远通信或第三方服务器（包括运营商、芯片供应商或您指定的服务器）。移远通信严格遵守相关法律法规，仅为实现产品功能之目的或在适用法律允许的情况下保留、使用、披露或以其他方式处理相关数据。当您与第三方进行数据交互前，请自行了解其隐私保护和数据安全政策。

## 免责声明

- 1) 移远通信不承担任何因未能遵守有关操作或设计规范而造成损害的责任。
- 2) 移远通信不承担因本文档中的任何因不准确、遗漏、或使用本文档中的信息而产生的任何责任。
- 3) 移远通信尽力确保开发中功能的完整性、准确性、及时性，但不排除上述功能错误或遗漏的可能。除非另有协议规定，否则移远通信对开发中功能的使用不做任何暗示或法定的保证。在适用法律允许的最大范围内，移远通信不对任何因使用开发中功能而遭受的损害承担责任，无论此类损害是否可以预见。
- 4) 移远通信对第三方网站及第三方资源的信息、内容、广告、商业报价、产品、服务和材料的可访问性、安全性、准确性、可用性、合法性和完整性不承担任何法律责任。

版权所有 © 上海移远通信技术股份有限公司 2024，保留一切权利。

**Copyright © Quectel Wireless Solutions Co., Ltd. 2024.**

# 文档历史

## 修订记录

版本	日期	作者	变更描述
-	2023-12-06	Aaron TIAN	文档创建
1.0	2024-07-10	Tommy TANG	受控版本

## 目录

文档历史 .....	3
目录 .....	4
图片索引 .....	5
<b>1 引言 .....</b>	<b>6</b>
<b>2 摄像头模组摆放位置.....</b>	<b>7</b>
<b>3 摄像头模组厂商提供的信息.....</b>	<b>9</b>
<b>4 摄像头硬件电路原理图.....</b>	<b>10</b>
<b>5 添加传感器驱动.....</b>	<b>12</b>
5.1. 代码路径 .....	12
5.2. 添加传感器驱动步骤.....	12
5.2.1. 配置硬件电路连接 .....	13
5.2.2. 配置驱动参数.....	18
5.2.2.1. 添加传感器驱动 .....	18
5.2.2.2. 配置传感器驱动 .....	19
5.2.2.3. 配置上下电时序 .....	20
5.2.3. 配置 Chromatix 效果参数 .....	23
<b>6 添加自动对焦马达驱动.....</b>	<b>24</b>
6.1. 更新设备树 .....	24
6.2. 添加自动对焦马达驱动参数文件 .....	25
<b>7 摄像头功能验证.....</b>	<b>26</b>
7.1. 预览 .....	26
7.2. 快照 .....	26
7.3. 视频 .....	26
<b>8 摄像头状态及异常排查.....</b>	<b>27</b>
8.1. 摄像头传感器运行失败 .....	27
8.2. MIPI 异常.....	27
<b>9 附录 术语缩写.....</b>	<b>28</b>

## 图片索引

图 1: 摄像头模组向安装 .....	7
图 2: 摄像头模组向安装 .....	7
图 3: 标准读数中的物体、传感器和显示的图像 .....	8
图 4: 读数顺序和显示的图像 .....	8
图 5: 后置主摄像头 EVB 硬件电路连接原理图 .....	10

# 1 引言

本文档以移远通信配套 EVB 开发板上的后置主摄像头 S5K4H7 为例介绍如何在移远通信 QSM368ZP-WF 的 Linux 系统上及移远通信 SG368Z 系列模块的 Linux 和 Ubuntu 系统上点亮摄像头模组相关器件。

摄像头模组驱动调试主要包括以下部分：

- 传感器驱动
- 自动对焦马达驱动





## 备注

1. 上述摄像头模组设计原理图仅作为示例参考，具体原理图请从相应的摄像头模组应商处获取。
2. 摄像头安装角度可以通过修改摄像头模组配置文件 `kernel/drivers/media/i2c/s5k4h7.c` 中的 `S5K4H7_REG_ORIENTATION_MODE` 进行调整。以主摄像头 S5K4H7 为例：

```
#define S5K4H7_REG_ORIENTATION_MODE 0x0101
```

- 如果摄像头模组位置相比图1或图2所示相差 180 度，即小人图像的头部对着显示屏的长边，则最终图像显示会相差 180 度。此时，对于后摄需将角度调整为 270 度，对于前摄需将角度调整为 90 度。
  - 如果摄像头模组位置相比图1或图2所示相差 90 度，则无法通过软件配置进行调整的，只能通过重新打样摄像头模组来解决。
3. 如果摄像头模组位置相比图1或图2所示相差 180 度，可以通过修改传感器的 `mirror` 寄存器和 `flip` 寄存器进行旋转。假设当前 `mirror` 寄存器和 `flip` 寄存器的值为 `0x0`，将两个寄存器设置为 `0x1`，即可实现 180 度旋转。

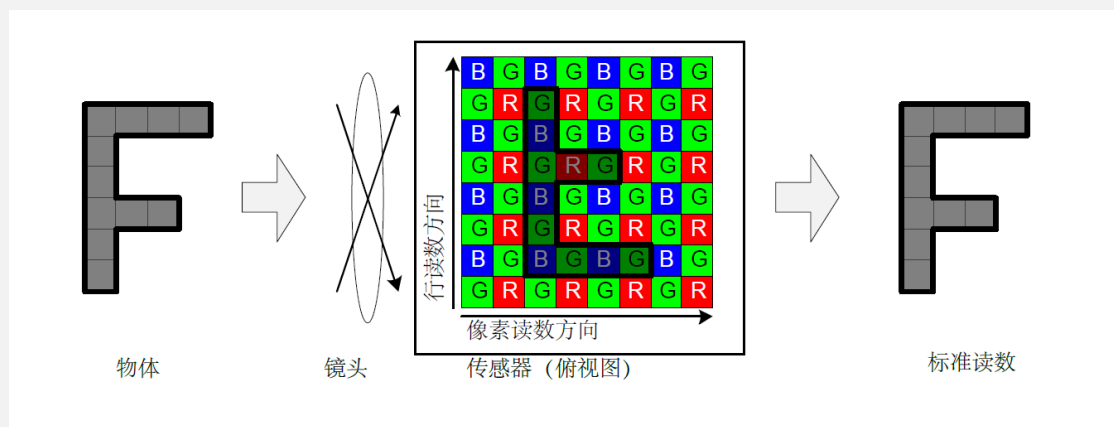


图 3：标准读数中的物体、传感器和显示的图像

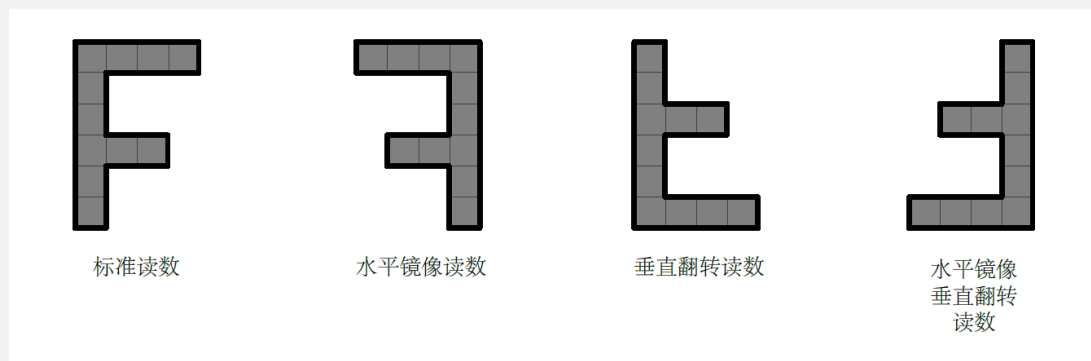


图 4：读数顺序和显示的图像

## 3 摄像头模组厂商提供的信息

摄像头模组厂商/IC 厂商应提供如下信息：

1. 摄像头模组规格书和传感器 IC 数据手册。若摄像头有自动对焦功能，还需提供对焦马达驱动 IC 数据手册。
2. 传感器的驱动代码（若该代码经过移远通信验证，则将由移远通信提供。摄像头模组厂商至少要提供寄存器配置，移远通信可协助重新创建驱动，但调试时间需要增加，用户也可以参考平台已有的驱动自行创建）。
3. 传感器的效果参数代码（摄像头点亮时，可临时使用移远通信已有的效果参数，等点亮以后，可再根据具体模块进行重新配置）。
4. 若摄像头模组带有对焦马达，需提供马达的用户空间驱动代码或初始化代码以及马达驱动 IC 厂商验证过的寄存器。如果对焦马达型号是 QSM368ZP-WF 或 SG368Z 系列模块支持的，则可以直接使用模块中自带的驱动代码。

### 备注

1. 建议优先寻求供应商支持进行效果优化，若供应商无法支持，可联系移远通信技术支持沟通是否可以协助优化。
2. 移远通信可协助添加马达驱动，但无法协助配置马达效果，马达效果请联系供应商协助。

## 4 摄像头硬件电路原理图

下图为 QSM368ZP-WF 和 SG368Z 系列模块连接到对应 EVB 开发板后置主摄像头的电路原理图。

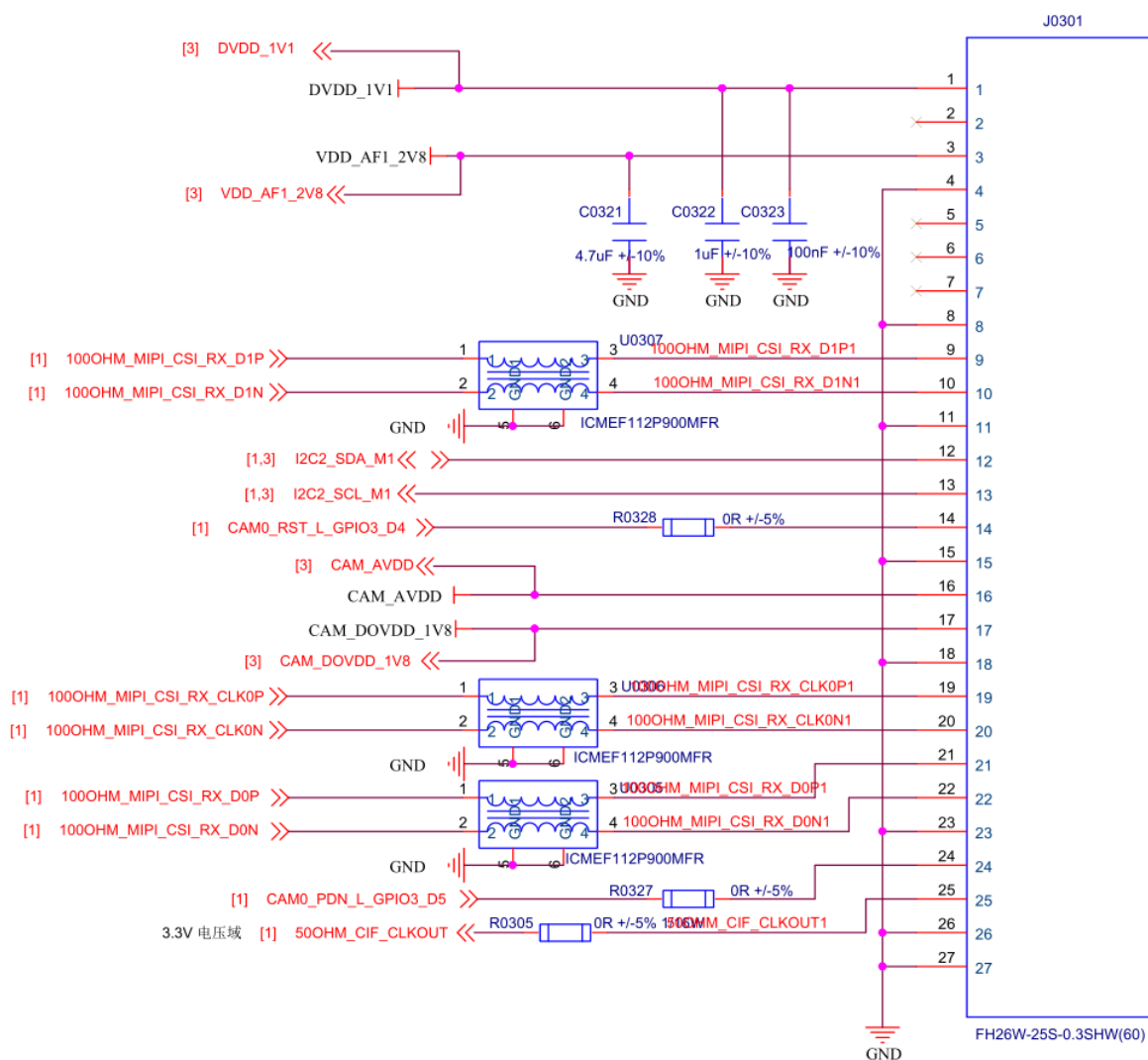


图 5：后置主摄像头 EVB 硬件电路连接原理图

- 供电：AVDD、DOVDD（IOVDD）、DVDD、AF\_VDD
- 复位：RST
- 摄像头关断：PWDN
- 时钟：MCLK
- MIPI数据：MDP0 MDN0、MDP1 MDN1、MDP2 MDN2、MDP3 MDN3
- MIPI时钟：MCP MCN
- I2C：SDA、SCL

## 备注

1. 摄像头传感器的主要供电引脚是 AVDD（2.8V）、IOVDD（1.8V）和 DVDD（1.2V）。AF\_VDD（2.8V）用于对焦马达供电，存在马达的时候才需要配置。通常情况下，电压值如上述括号中标识，具体的电压值需要查阅所使用的传感器 IC 数据手册。
2. 具体的引脚连接方式，请查看摄像头硬件电路连接原理图。如果供电、MCLK、I2C、RST、PWDN 引脚和上电时序配置正确，则摄像头传感器可正常工作。在开机过程中，QSM368ZP-WF 和 SG368Z 系列模块会通过 I2C 与传感器通信，并读取传感器的 ID 与驱动中配置的 ID 进行匹配，如果匹配成功，则认为该传感器注册成功。

## 5 添加传感器驱动

### 5.1. 代码路径

对于 QSM368ZP-WF 和 SG368Z 系列模块，摄像头配置相关文件路径如下：

1. 硬件相关配置路径：

```
kernel/arch/arm64/boot/dts/rockchip/rk3568.dtsi
kernel/arch/arm64/boot/dts/rockchip/rk3568-camera.dtsi
kernel/arch/arm64/boot/dts/rockchip/rk3568-evb1-ddr4-v10.dtsi
kernel/arch/arm64/boot/dts/rockchip/rk3568-pinctrl.dtsi
```

2. 摄像头传感器驱动参数文件路径：

```
kernel/drivers/media/i2c/s5k4h7.c
```

3. 摄像头传感器效果参数文件路径：

```
external/camera_engine_rkaiq/iqfiles/isp21/s5k4h7_SA80-00N8-0_default.json
```

4. so 库配置文件：

```
kernel/drivers/media/i2c/Makefile
kernel/drivers/media/i2c/Kconfig
kernel/arch/arm64/configs/rockchip_linux_defconfig
```

### 5.2. 添加传感器驱动步骤

**步骤 1：**配置硬件电路连接；

**步骤 2：**配置驱动参数；

**步骤 3：**配置 Chromatix 效果参数。

### 5.2.1. 配置硬件电路连接

摄像头模组的硬件电路配置在.dtsi 设备树文件中配置，文件路径如下：

*kernel/arch/arm64/boot/dts/rockchip/rk3568-camera.dtsi*

以后置主摄像头 S5K4H7 为例，其设备树文件配置如下：

```
s5k4h7: s5k4h7@2d {
    compatible = "quec,s5k4h7";
    status = "okay";
    reg = <0x2d>;
    power-domains = <&power RK3568_PD_VI>;
    dvdd-supply = <&vcc_dvdd>;
    avdd-supply = <&vcc_avdd>;
    iovdd-supply = <&vcc_iovdd>;

    clocks = <&cru CLK_CIF_OUT>;
    clock-names = "xvclk";
    pinctrl-names = "default","rockchip,camera_default","rockchip,camera_sleep";
    pinctrl-0 = <&cif_clk>;
    pinctrl-1 = <&rear_reset_active>;
    pinctrl-2 = <&rear_reset_sleep>;
    reset-gpios = <&gpio3 RK_PD4 GPIO_ACTIVE_HIGH>;
    pwn-gpios = <&gpio3 RK_PD5 GPIO_ACTIVE_HIGH>;

    rockchip,camera-module-index = <0>;
    rockchip,camera-module-facing = "back";
    rockchip,camera-module-name = "SA80-00N8-0";
    rockchip,camera-module-lens-name = "default";
    port {
        cam_out0: endpoint {
            remote-endpoint = <&dphy1_in>;
            data-lanes = <1 2>;
        };
    };
};
```

#### 1. 配置后置主摄像头 S5K4H7 属性：

```
s5k4h7: s5k4h7@2d {
    compatible = "quec,s5k4h7";
    status = "okay";
    reg = <0x2d>;
    rockchip,camera-module-index = <0>;
```

```

        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "SA80-00N8-0";
        rockchip,camera-module-lens-name = "default";

port {
    ucam_out0: endpoint {
        remote-endpoint = <&dphy0_in>;
        data-lanes = <1 2 3 4>;
    };
};

```

参数	描述
<i>s5k4h7@2d</i>	<i>s5k4h7</i> 为传感器名称， <i>2d</i> 为传感器 7 位 I2C 地址
<i>rockchip,camera-module-index</i>	摄像头模组编号，指定摄像头使用的是哪一组硬件配置
<i>rockchip,camera-module-facing</i>	摄像头模组的方向，取值为" <b>black</b> "或" <b>front</b> "
<i>rockchip,camera-module-name</i>	传感器模块的名称。调试文件名由传感器型号、传感器模组名称和镜头名称组成
<i>rockchip,camera-module-lens-name</i>	传感器镜头的名称。调试文件名由传感器型号、传感器模组名称和镜头名称组成
<i>remote-endpoint</i>	MIPI DPHY 端的端口名称
<i>data-lanes</i>	MIPI lane 数量，1 个 lane 为<1>，4 个 lane 为<1 2 3 4>

## 2. 配置 regulator 电源控制:

```

power-domains = <&power RK3568_PD_VI>;
dvdd-supply = <&vcc_dvdd>;
avdd-supply = <&vcc_avdd>;
iovdd-supply = <&vcc_iovdd>;

```

参数	描述
<i>power-domains</i>	使能传感器相关电源域
<i>dvdd-supply</i>	DVDD
<i>avdd-supply</i>	AVDD
<i>iovdd-supply</i>	IOVDD

### 3. 配置 GPIO:

```
pinctrl-names = "default","rockchip,camera_default","rockchip,camera_sleep";
pinctrl-0 = <&cif_clk>;
pinctrl-1 = <&rear_reset_active>;
pinctrl-2 = <&rear_reset_sleep>;
reset-gpios = <&gpio3 RK_PD4 GPIO_ACTIVE_HIGH>;
pwn-gpios = <&gpio3 RK_PD5 GPIO_ACTIVE_HIGH>;
```

参数	描述
<i>reset-gpios</i>	传感器的复位引脚
<i>pwn-gpios</i>	传感器的关断引脚

*pinctrl-0*、*pinctrl-1* 和 *pinctrl-2* 对应 GPIO 的三种状态。

*gpios* 中描述使用到了哪些 GPIO。如果修改了 *gpios* 中的引脚, *rear\_reset\_active* 和 *rear\_reset\_sleep* 也要相应修改。*rear\_reset\_active* 和 *rear\_reset\_sleep* 在文件 *kernel/arch/arm64/boot/dts/rockchip/rk3568-camera.dtsi* 中定义, 如下所示:

```
rear_reset_active: rear-reset-active {
    rockchip,pins =
        <3 RK_PD4 RK_FUNC_GPIO &pcfg_pull_up>,
        <3 RK_PD5 RK_FUNC_GPIO &pcfg_pull_up>;
};
rear_reset_sleep: rear-reset-sleep {
    rockchip,pins =
        <3 RK_PD4 RK_FUNC_GPIO &pcfg_pull_none>,
        <3 RK_PD5 RK_FUNC_GPIO &pcfg_pull_none>;
};
```

### 4. 配置时钟源:

```
clocks = <&cru CLK_CIF_OUT>;
clock-names = "xvclk";
```

参数	描述
<i>clocks</i>	传感器主时钟源配置, 摄像头有 4 个时钟源: <&pmucru CLK_WIFI> <&cru CLK_CIF_OUT> <&cur CLK_CAM0_OUT> <&cur CLK_CAM1_OUT>



*clock-names*

该值固定为 *xvclk*（驱动会选取一个名为 *xvclk* 的时钟），即 24 M 时钟

需要和 *gpios* 中的主时钟引脚对应。如果修改了 *gpios* 中主时钟引脚，此处也需要变更为对应的时钟源配置。除此之外不建议修改。

## 5. 配置 CSI 总线：

QSM368ZP-WF 和 SG368Z 系列模块仅有一个标准物理 MIPI CSI2 DPHY，可以在 Full Mode 和 Split Mode 两种模式下工作，可拆分为三个逻辑 DPHY，分别是 *csi2\_dphy0*、*csi2\_dphy1* 和 *csi2\_dphy2*（参见 *kernel/arch/arm64/boot/dts/rockchip/rk3568.dtsi*），其中 *csi2\_dphy0* 和 *csi2\_dphy1* 或 *csi2\_dphy2* 互斥，不可同时使用。

### ● Full Mode

- 仅使用 *csi2\_dphy0*，*csi2\_dphy1* 或 *csi2\_dphy2* 不可使用；
- 最多支持 4 个数据 lane；
- 最大速率为 2.5 Gbps/lane。

### ● Split Mode

- 仅使用 *csi2\_dphy1* 和 *csi2\_dphy2*，*csi2\_dphy0* 不可使用；
- *csi2\_dphy1* 和 *csi2\_dphy2* 可同时使用；
- *csi2\_dphy1* 和 *csi2\_dphy2* 各自最多支持 2 个数据 lane；
- *csi2\_dphy1* 对应物理 DPHY 的 lane0 或 lane1；
- *csi2\_dphy2* 对应物理 DPHY 的 lane2 或 lane3；
- 最大速率为每 2.5 Gbps/lane。

ISP 链路关系：传感器 → *csi2\_dphy0* → ISP

```
&csi2_dphy0 {
    status = "okay";
    ports {
        #address-cells = <1>;
        #size-cells = <0>;
        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;
            dphy0_in: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&cam_out0>;
                data-lanes = <1 2>;
            };
        };
        port@1 {
```

```

        reg = <1>;
        #address-cells = <1>;
        #size-cells = <0>;
        dphy0_out: endpoint@1 {
            reg = <1>;
            remote-endpoint = <&isp0_in>;
        };
    };
};

```

参数	描述
<i>csi2_dphy0</i>	<i>csi2_dphy0</i> 与 <i>csi2_dphy1</i> 或 <i>csi2_dphy2</i> 互斥，不能同时使用
<i>port@0</i>	<i>csi2_dphy0</i> 的入口端口
<i>port@1</i>	<i>csi2_dphy0</i> 的导出端口
<i>remote-endpoint</i>	对应的入口或导出端口节点 <ul style="list-style-type: none"> <li>● <i>port@0</i> 对应传感器端口的端口名称</li> <li>● <i>port@1</i> 对应 ISP 端口的端口名称</li> </ul>
<i>data-lanes</i>	CSI2 DPHY lane 的数量，1 个 lane 表示为<1>，4 个 lane 表示为<1 2 3 4>，须与传感器端一致。

```

&rkisp_vir0 {
    status = "okay";
    port {
        #address-cells = <1>;
        #size-cells = <0>;
        isp0_in: endpoint@0 {
            reg = <0>;
            remote-endpoint = <&dphy0_out>;
        };
    };
};

```

参数	描述
<i>rkisp_vir0</i>	ISP 节点
<i>port</i>	<i>rkisp_vir0</i> 的入口端口
<i>remote-endpoint</i>	<i>csi2_dphy</i> 端的端口名称

## 5.2.2. 配置驱动参数

摄像头传感器的驱动参数文件主要包含传感器初始化寄存器和参数信息，如尺寸、数据格式等。通常需要摄像头模组厂商提供。如果传感器驱动经过移远通信验证，则可以由移远通信提供。如果摄像头模组厂商也没有对应的驱动配置文件，至少需要提供寄存器配置和数据手册，移远通信可以基于此协助配置驱动参数文件。

### 5.2.2.1. 添加传感器驱动

将驱动参数文件添加至如下路径中：*kernel/drivers/media/i2c/*

以配套 EVB 开发板后置主摄像头 S5K4H7 为例，该摄像头驱动参数文件参考路径如下：

```
kernel/arch/arm64/configs/rockchip_linux_defconfig
kernel/drivers/media/i2c/
├── s5k4h7.c
├── Makefile
└── Kconfig
```

配置 *kernel/drivers/media/i2c/KConfig* 自定义编译对象。

```
config VIDEO_S5K4H7
    tristate "Samsung S5K4H7 sensor support"
    depends on MEDIA_CAMERA_SUPPORT
    depends on I2C && VIDEO_V4L2 && VIDEO_V4L2_SUBDEV_API
    help
        This is a V4L2 sensor driver for Samsung S5K4H7 raw
        camera sensor.
```

配置 *kernel/drivers/media/i2c/Makefile* 添加编译条件。

```
obj-$(CONFIG_VIDEO_S5K4ECGX) += s5k4ecgx.o
obj-$(CONFIG_VIDEO_S5K5BAF) += s5k5baf.o
obj-$(CONFIG_VIDEO_S5KGM1SP) += s5kgm1sp.o
obj-$(CONFIG_VIDEO_S5K4H7) += s5k4h7.o
obj-$(CONFIG_VIDEO_S5K4H7B) += s5k4h7b.o
obj-$(CONFIG_VIDEO_S5C73M3) += s5c73m3/
obj-$(CONFIG_VIDEO_AD1653) += adp1653.o
obj-$(CONFIG_VIDEO_LM3560) += lm3560.o
obj-$(CONFIG_VIDEO_LM3646) += lm3646.o
obj-$(CONFIG_VIDEO_SGM3784) += sgm3784.o
obj-$(CONFIG_VIDEO_SMIAPP_PLL) += smiapp-pll.o
obj-$(CONFIG_VIDEO_AK881X) += ak881x.o
obj-$(CONFIG_VIDEO_IR_I2C) += ir-kbd-i2c.o
```

修改 `kernel/arch/arm64/configs/rockchip_linux_defconfig` 确保所有新添加的库都可以打包到镜像中。

```
CONFIG_VIDEO_TC35874X=y
CONFIG_VIDEO_RK628_CSI=y
CONFIG_VIDEO_LT6911UXC=y
CONFIG_VIDEO_LT8619C=y
CONFIG_VIDEO_OS04A10=y
CONFIG_VIDEO_OV4689=y
CONFIG_VIDEO_OV5695=y
CONFIG_VIDEO_OV7251=y
CONFIG_VIDEO_OV13850=y
CONFIG_VIDEO_GC8034=y
CONFIG_VIDEO_S5K4H7=y
CONFIG_VIDEO_S5K4H7B=y
```

#### 5.2.2.2. 配置传感器驱动

添加传感器驱动后，可以根据需要对驱动文件 `kernel/drivers/media/i2c/s5k4h7.c` 进行配置，重要参数配置说明如下。一般来说，移远通信提供的驱动不需要修改配置。

1. 配置 I2C 地址和传感器 ID。驱动上电后会读取传感器 ID。

```
#define CHIP_ID                0x487B
#define S5K4H7_REG_CHIP_ID    0x0000 //read only reg
```

2. 配置传感器输出。

```
#define S5K4H7_LANES          2
#define S5K4H7_BITS_PER_SAMPLE 10
```

### 3. 配置寄存器。

```
static const struct s5k4h7_mode supported_modes[] = {
    {
        .width = 3264,
        .height = 2448,
        .max_fps = {
            .numerator = 10000,
            .denominator = 300000,
        },
        .exp_def = 0x0990,
        .hts_def = 0x0DFC,
        .vts_def = 0x09E2,
        .reg_list = s5k4h7_linear_preview_regs,
        .hdr_mode = NO_HDR,
        .vc[PAD0] = V4L2_MBUS_CSI2_CHANNEL_0,
    },
    {
        .width = 3264,
        .height = 2448,
        .max_fps = {
            .numerator = 10000,
            .denominator = 300000,
        },
        .exp_def = 0x0990,
        .hts_def = 0x0DFC,
        .vts_def = 0x09E2,
        .reg_list = s5k4h7_linear_snapshot_regs,
        .hdr_mode = NO_HDR,
        .vc[PAD0] = V4L2_MBUS_CSI2_CHANNEL_0,
    },
};
```

#### 5.2.2.3. 配置上下电时序

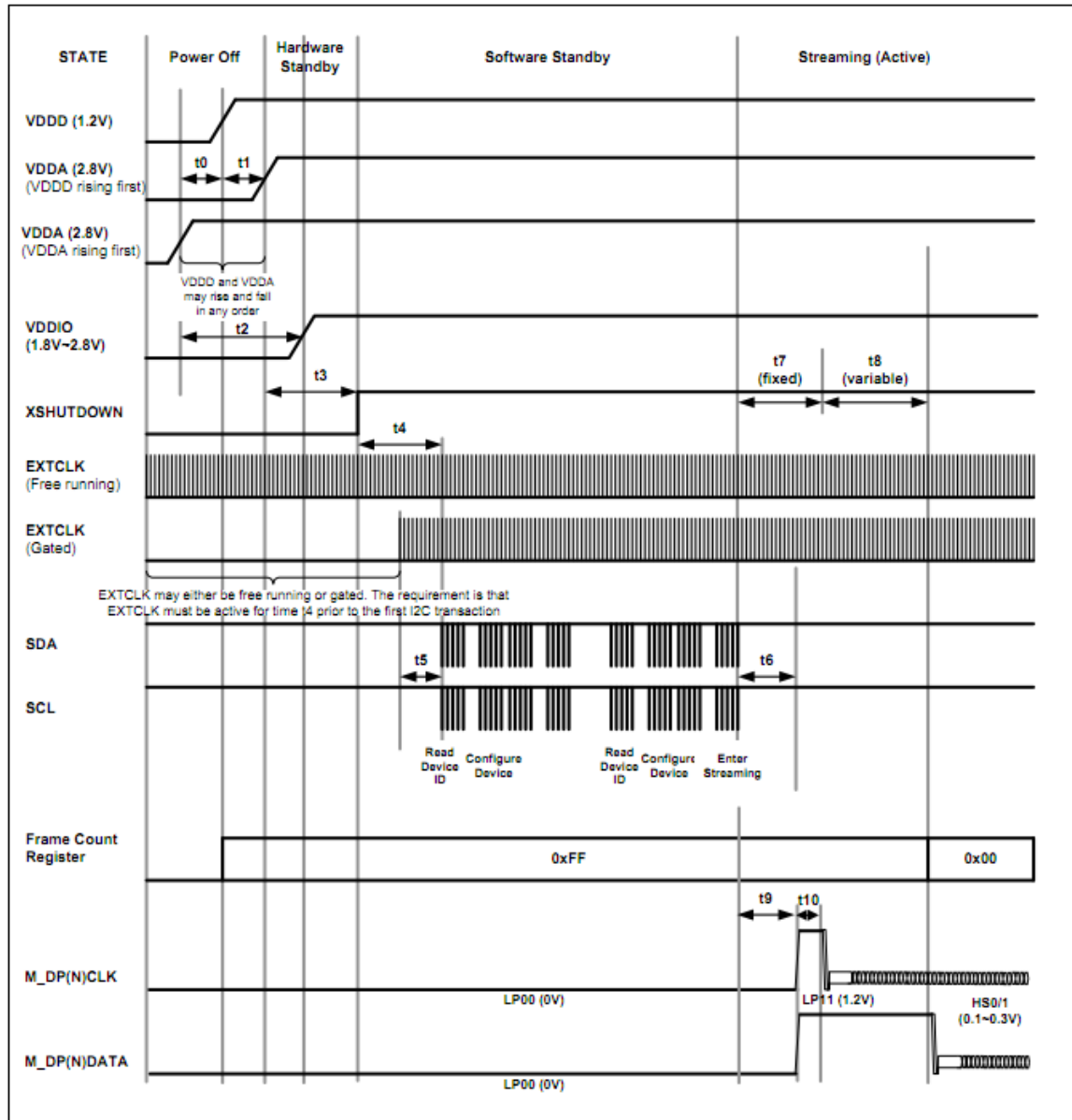
以配套 EVB 开发板后置主摄像头 S5K4H7 为例。S5K4H7 数据手册中描述的上电时序如下：

数字模拟电源电压和模拟电源电压可以按任意顺序上电，例如先上电 VDDD，后上电 VDDA，或先上电 VDDA，后上电 VDDD。

上电：

- 如果电源接通时 XSHUTDOWN 为低电平，则传感器模块进入硬件待机模式。
- 如果电源接通时 XSHUTDOWN 为高电平，则传感器模块进入软件待机模式。

以上两种情况下，芯片上的上电复位单元确保了正确初始化 I2C 寄存器的值为默认值。EXTCLK 时钟可以在软件待机模式期间先处于低电平状态，然后被启用，也可以在 VDDIO 供电时自由运行。



配置路径: `kernel/drivers/media/i2c/s5k4h7.c`

```
static int __s5k4h7_power_on(struct s5k4h7 *s5k4h7)
{
    int ret;
    u32 delay_us;
    struct device *dev = &s5k4h7->client->dev;

    if (!IS_ERR_OR_NULL(s5k4h7->pins_default)) {
        ret = pinctrl_select_state(s5k4h7->pinctrl,
                                   s5k4h7->pins_default);
        if (ret < 0)
            dev_err(dev, "could not set pins\n");
    }
    ret = clk_set_rate(s5k4h7->xvclk, S5K4H7_XVCLK_FREQ);
    if (ret < 0)
        dev_warn(dev, "Failed to set xvclk rate (24MHz)\n");
    if (clk_get_rate(s5k4h7->xvclk) != S5K4H7_XVCLK_FREQ)
        dev_warn(dev, "xvclk mismatched, modes are based on 24MHz\n");
    ret = clk_prepare_enable(s5k4h7->xvclk);

    gpiod_set_value_cansleep(s5k4h7->pwdn_gpio, 0);
    gpiod_set_value_cansleep(s5k4h7->reset_gpio, 0);
    clk_set_rate(s5k4h7->xvclk, 0);
    usleep_range(1000, 1100);

    ret = regulator_enable(s5k4h7->iovdd_regulator);
    ret = regulator_enable(s5k4h7->dvdd_regulator);
    ret = regulator_enable(s5k4h7->avdd_regulator);
    usleep_range(1000, 1100);
    clk_set_rate(s5k4h7->xvclk, S5K4H7_XVCLK_FREQ);
    gpiod_set_value_cansleep(s5k4h7->pwdn_gpio, 1);
    gpiod_set_value_cansleep(s5k4h7->reset_gpio, 1);
    usleep_range(10 * 1000, 11 * 1000);
    /* 8192 cycles prior to first SCCB transaction */
    delay_us = s5k4h7_cal_delay(8192);
    usleep_range(delay_us, delay_us * 2);

    return 0;
}
```

```
static void __s5k4h7_power_off(struct s5k4h7 *s5k4h7)
{
    int ret;
    /* 512 xvclk cycles after the last SCCB transaction or MIPI frame end */
    usleep_range(30, 50);

    gpiod_set_value_cansleep(s5k4h7->reset_gpio, 0);
    gpiod_set_value_cansleep(s5k4h7->pwdn_gpio, 0);
    clk_disable_unprepare(s5k4h7->xvclk);
    if (!IS_ERR_OR_NULL(s5k4h7->pins_sleep)) {
        ret = pinctrl_select_state(s5k4h7->pinctrl,
                                   s5k4h7->pins_sleep);
        if (ret < 0)
            dev_dbg(&s5k4h7->client->dev, "could not set pins\n");
    }
    usleep_range(1*1000, 2 * 1000);
    regulator_disable(s5k4h7->avdd_regulator);
    regulator_disable(s5k4h7->dvdd_regulator);
    regulator_disable(s5k4h7->iovdd_regulator);
}
```

根据传感器数据手册要求修改上下电时序。

### 5.2.3. 配置 Chromatix 效果参数

效果参数文件位于 `external/camera_engine_rkaiq/iqfiles/isp21/`中。

`iqfiles` 目录存储传感器的 `iq` 参数和 `xml` 文件。文件需要按如下规则命名：

`Aaa_xxx-yyy.jsonn` 和 `Aaa_xxx-yyy.xml`，其中：

`Aaa`: 传感器型号

`xxx`: 传感器模组名称

`yyy`: 镜头名称

例如：

`s5k4h7_SA80-00N8-0_default.json`

`s5k4h7_SA80-00N8-0_default.xml`



## 6 添加自动对焦马达驱动

如果摄像头模组具有自动对焦功能，则需要调试自动对焦驱动。以下均为示例内容，介绍如何配置自动对焦功能。

### 6.1. 更新设备树

更新设备树节点。在 `kernel/arch/arm64/boot/dts/rockchip/rk3568-camera.dtsi` 中为马达节点添加一个条目，并将 `dw9714` 分配为该马达的节点，如下所示：

```
dw9714: dw9714@0c {
    compatible = "quec,dw9714";
    status = "okay";
    reg = <0x0c>;
    rockchip,vcm-start-current = <0>;
    rockchip,vcm-rated-current = <100>;
    rockchip,vcm-step-mode = <4>;
    rockchip,camera-module-index = <0>;
    rockchip,camera-module-facing = "back";
};
s5k4h7: s5k4h7@2d {
    .....
    lens-focus = <&dw9714>;
    .....
};
```

参数	描述
<code>rockchip,vcm-start-current</code>	VCM 的起动电流。VCM 恰好能够推动摄像头模组镜头从可移动行程的最近端（摄像头模组的远焦点）移动时，软件定义当前 VCM 驱动 IC 的输出电流值为起动电流
<code>rockchip,vcm-rated-current</code>	VCM 的额定电流。VCM 恰好将摄像头模组镜头推动到可移动行程的最远端（摄像头模组的近焦点）时，软件定义当前 VCM 驱动 IC 的输出电流值为额定电流
<code>rockchip,vcm-step-mode</code>	VCM 的电流输出模式。VCM 移动过程中产生振荡。需要考虑 VCM 驱动 IC 电流输出的变化对 VCM 振荡周期的影响，以便最小化振荡。输出模式决定输出电流变化到目标值的时间

<code>rockchip,camera-module-index</code>	摄像头模组编号
<code>rockchip,camera-module-facing</code>	摄像头模组的方向，取值为"front"和"back"

## 6.2. 添加自动对焦马达驱动参数文件

配置 `kernel/drivers/media/i2c/KConfig` 自定义编译对象。

```
config VIDEO_DW9714
    tristate "DW9714 lens voice coil support"
    depends on I2C && VIDEO_V4L2 && MEDIA_CONTROLLER
    depends on VIDEO_V4L2_SUBDEV_API
    ---help---
    This is a driver for the DW9714 camera lens voice coil.
    DW9714 is a 10 bit DAC with 120mA output current sink
    capability. This is designed for linear control of
    voice coil motors, controlled via I2C serial interface.
```

配置 `kernel/drivers/media/i2c/Makefile` 添加编译条件。

```
obj-$(CONFIG_VIDEO_SAA7185) += saa7185.o
obj-$(CONFIG_VIDEO_SAA6752HS) += saa6752hs.o
obj-$(CONFIG_VIDEO_AD5820) += ad5820.o
obj-$(CONFIG_VIDEO_AK7375) += ak7375.o
obj-$(CONFIG_VIDEO_DW9714) += dw9714.o
obj-$(CONFIG_VIDEO_DW9718) += dw9718.o
obj-$(CONFIG_VIDEO_DW9807_VCM) += dw9807-vcm.o
obj-$(CONFIG_VIDEO_FP5501) += fp5501.o
obj-$(CONFIG_VIDEO_FP5510) += fp5510.o
obj-$(CONFIG_VIDEO_GT9760S) += gt9760s.o
obj-$(CONFIG_VIDEO_VM149C) += vm149c.o
obj-$(CONFIG_VIDEO_ADV7170) += adv7170.o
obj-$(CONFIG_VIDEO_ADV7175) += adv7175.o
obj-$(CONFIG_VIDEO_ADV7180) += adv7180.o
obj-$(CONFIG_VIDEO_ADV7183) += adv7183.o
```

# 7 摄像头功能验证

执行 **Gstreamer** 命令验证摄像头是否正常工作。若摄像头正常工作，则显示如下内容。

## 7.1. 预览

```
gst-launch-1.0 -e v4l2src device=/dev/video0 ! video/x-raw, width=640, height=480, format=Nv12, framerate=30/1 ! decodebin ! videoconvert ! autovideosink
```

## 7.2. 快照

```
gst-launch-1.0 -e v4l2src device=/dev/video0 num-buffers=1 ! video/x-raw,format=Nv12,width=640,height=480 ! mppjpegenc ! filesink location=/data/camera/cam0_640x480.jpg
```

## 7.3. 视频

```
gst-launch-1.0 -e v4l2src device=/dev/video0 ! video/x-raw,format=Nv12, width=640,height=480, framerate=30/1 ! mpph264enc ! queue ! h264parse ! qtmux ! filesink
```

## 8 摄像头状态及异常排查

若以上章节内容配置正确，摄像头传感器就可以正常工作。开机过程中，QSM368ZP-WF 和 SG368Z 系列模块会通过 I2C 与传感器进行通信，读取传感器中的 ID 与驱动中配置的 ID 进行匹配，若匹配成功，则摄像头注册成功，将显示类似如下日志信息：

```
s5k4h7: 2-002d: s5k4h7 probe success, sensor id: 0X487b
Rockchip-csi2-dphy csi2-dphy1: dphy1 match m00_b_s5k4h7 2-002d:bus type 4
```

### 8.1. 摄像头传感器运行失败

如果摄像头传感器无法与设备通信，则说明 I2C 地址错误或设备没有工作，日志信息显示如下：

```
s5k4h7: 2-002d: Unexpected sensor id(0), ret(-22)
```

此时，需要检查 I2C 地址、上电时序、设备树引脚配置等。

### 8.2. MIPI 异常

若摄像头已经注册成功，但是系统相机打开后出现黑屏，可能是因为 MIPI 信号异常。

MIPI 信号异常可能的原因如下：

1. 摄像头传感器本身没有出帧，需要检查传感器寄存器配置或者寄存器是否写入正确。
2. MIPI lane 配置错误。
3. 供电及 MCLK 不稳定或不符合传感器工作时的要求。需要参考传感器数据手册逐一检查。
4. 硬件连接问题导致 MIPI 传输过程中出现异常。

如果出现 sof freeze 异常，请参考以上可能的原因点进行排查。

## 9 附录 术语缩写

表 1：术语缩写

缩写	英文描述	中文描述
AF	Auto Focus	自动对焦
AF_VDD	Auto Focus Power Supply Voltage	自动对焦电路电源电压
API	Application Programming Interface	应用程序接口
AVDD	Analog Circuit Power Supply Voltage	模拟电路电源电压
AFVDD	Auto Focus Power Supply Voltage	自动对焦电路电源电压
CLK	Clock	时钟
CPP	Camera Post Processor	摄像机后置处理器
CSI	Camera Serial Interface	摄像机串行接口
CSID	Camera Serial Interface Decoder	摄像机串行接口解码器
DDR	Double Data Rate	双倍数据传输率
DOVDD (IOVDD)	Digital I/O Circuit Power Supply Voltage	数字量 I / O 电路电源电压
DVDD	Digital Core Circuit Power Supply Voltage	数字核心电路电源电压
EEPROM	Electrically Erasable Programmable Read Only Memory	带电可擦可编程只读存储器
EVB	Evaluation Board	评估板
GPIO	General-Purpose Input/Output	通用输入/输出口
HAL	Hardware Abstraction Layer	硬件抽象层
IC	Integrated Circuit	集成电路
I2C	Inter-Integrated Circuit	两线时串行总线

ISP	Image Signal Processor	图像处理
LDO	Low Dropout Regulator	低压差线性稳压器
LED	Light Emitting Diode	发光二极管
MCLK	Main Clock	主时钟
MIPI	Mobile Industry Processor Interface	移动产业处理器接口
OEM	Original Equipment Manufacture	原始设备厂商
OTP	One Time Programmable	一次性编程
PDAF	Phase Detection Auto Focus	相位检测自动对焦
PHY	Physical Layer	物理层
PMIC	Power Management IC	电源管理集成电路
PWDN	Power Down	下电
RST	Reset	重置
SCL	Serial Clock	串行时钟信号线
SDA	Serial Data	串行数据信号线